**Git Student Exercise**

**Part 1: Git Basics – Version Control Essentials**

**Exercise 1: Setup Git**

* **Objective:** Configure your Git identity.
* **Commands:**

git config --global user.name "Your Name"

git config --global user.email "you@example.com"

git config --list

*Expected Result:* Git user identity is set.

**Exercise 2: Create a Local Git Repo**

* **Objective:** Initialize a new project.
* **Steps:**

mkdir my-git-app

cd my-git-app

git init

* *Expected Result:* .git directory created.

**Exercise 3: Add and Commit Files**

* **Steps:**

echo "Hello Git!" > index.txt

git status

git add index.txt

git commit -m "Initial commit"

* *Expected Result:* File committed.

**Part 2: Git Branching & Merging**

**Exercise 4: Create and Switch Branches**

* **Objective:** Work on a feature in a new branch.
* **Steps:**

git branch feature-1

git checkout feature-1

echo "New Feature" >> index.txt

git commit -am "Add feature 1"

* *Expected Result:* New branch created and updated.

**Exercise 5: Merge Branch into Main**

* **Steps:**

git checkout main

git merge feature-1

* *Expected Result:* feature-1 changes merged into main.

**Exercise 6: Resolve Merge Conflict**

* **Steps:**
  + Modify the same line in index.txt on two branches (main and feature-conflict)
  + Try merging feature-conflict into main.
  + Git will report a conflict.

git merge feature-conflict

* + Edit file manually to resolve conflict.
  + Then:

git add index.txt

git commit -m "Resolved merge conflict"

**Part 3: Remote Repositories (GitHub)**

**Exercise 7: Connect to GitHub**

* **Steps:**

git remote add origin https://github.com/your-username/my-git-app.git

git branch -M main

git push -u origin main

**Exercise 8: Clone and Pull**

* **Steps:**

git clone https://github.com/your-username/my-git-app.git

cd my-git-app

git pull origin main

**Part 4: Git Log and History**

**Exercise 9: View Commit Logs**

* **Commands:**

git log

git log --oneline --graph

**Exercise 10: Revert and Reset**

* **Objective:** Undo mistakes.
* **Steps:**

git revert <commit-hash>

git reset --hard <older-commit>

**Part 5: Sample Project: “Task Tracker App”**

**🌱 Step 1: Initialize Project**

mkdir task-tracker

cd task-tracker

git init

echo "# Task Tracker App" > README.md

git add .

git commit -m "Initialize project with README"

**🌱 Step 2: Add Main Code**

echo "function addTask(task) { console.log(task); }" > app.js

git add .

git commit -m "Add base task function"

**🌱 Step 3: Create Feature Branch**

git checkout -b ui-enhancement

echo "<h1>Task Tracker UI</h1>" > index.html

git add .

git commit -m "Add basic UI"

**🌱 Step 4: Merge to Main**

git checkout main

git merge ui-enhancement

**Git Cheat Sheet Summary**

| **Command** | **Description** |
| --- | --- |
| git init | Start a new Git repo |
| git add . | Stage changes |
| git commit -m "msg" | Save snapshot |
| git branch | List branches |
| git checkout <branch> | Switch branch |
| git merge <branch> | Merge another branch |
| git status | Show changes |
| git log | Show commit history |
| git remote add origin <url> | Link remote repo |
| git push | Push to remote |
| git pull | Pull from remote |
| git clone <url> | Clone repo |
| git reset --hard | Rollback commit |
| git revert <hash> | Undo a specific commit |

**Part 6: Intermediate Branching and Collaboration**

**Exercise 11: Delete a Branch**

* **Objective:** Clean up merged branches.

git branch -d feature-1 # delete locally

git push origin --delete feature-1 # delete from remote

**Exercise 12: Track Remote Branch**

* **Steps:**

git checkout -b dev origin/dev

git pull

**Exercise 13: Work with stash**

* **Scenario:** Save work-in-progress without committing.
* **Steps:**

echo "console.log('temp')" >> app.js

git stash

git status # Clean working tree

git stash pop # Restore stashed changes

**Exercise 14: Tagging Releases**

* **Steps:**

git tag v1.0

git push origin v1.0

**📌 Part 7: Working with .gitignore and Cleanup**

**Exercise 16: Use .gitignore**

* **Steps:**

echo "node\_modules/" > .gitignore

echo "\*.log" >> .gitignore

git add .gitignore

git commit -m "Add .gitignore file"

**Exercise 17: Clean Untracked Files**

* **Steps:**

git clean -n # Dry run

git clean -f # Actually remove

**Part 8: Rewriting History (Advanced Use)**

**Use with caution (not for public/shared branches)**

**Exercise 18: Amend Last Commit**

* **Steps:**

git commit --amend -m "Updated commit message"

**Exercise 19: Interactive Rebase (Squashing)**

* **Steps:**

git log --oneline # copy last 3 commit hashes

git rebase -i HEAD~3

# Use editor to mark two commits as 'squash'

**Part 9: GitHub Fork and Pull Request Flow**

**Exercise 20: Fork a Repo on GitHub**

* Go to any public GitHub repo and click **Fork**.

**Exercise 21: Clone Fork and Add Remote**

git clone https://github.com/your-username/sample-app.git

cd sample-app

git remote add upstream https://github.com/original-owner/sample-app.git

**Exercise 22: Keep Fork in Sync**

git fetch upstream

git checkout main

git merge upstream/main

git push origin main